Applying ML methods in HiggsML Binary Classification Challenge

Guosheng Feng, Haixin Shi, Zhiye Wang
Department of Computer Science, EPFL, Switzerland

***Abstract*—In this project, we focus on binary classification task on the real-world CERN particle accelerator dataset. We first conduct data analysis and clean the data by regrouping it, deleting meaningless columns in each groups, removing outliers, and performing normalization. Then, we expand features through arithmetical operations. Finally, we apply several machine learning models to fit the data and use cross-validation to generate the results. Hyper-parameters are obtained through grid-search. As a result, our best model could achieve 83.1% accuracy on the public testing set.**

I. Introduction

Through this work we study the ML models for dealing with the HiggsML Challenge task [1]. Specifically, HiggsML involves a simulated dataset from ATLAS experiment at CERN, which contains 250,000 data items each with 30 features that are divided into two classes. We first analyze the data and preprocess it by dealing with missing values, normalizing features, and expanding features. For model chose, we compare three basic models as linear regression, ridge regression and logistic regression, each equipped with l_2 regularization terms. All results are evaluated with cross-validations to avoid over-fitting. Consequently, we found data regrouping and feature expansion crucial for the result. Models are also sensitive to hyper-parameters (e.g. regularization weight, learning rate, polynomial degrees), thus we apply grid search with visualization for properly choosing hyper-parameters.

II. Data Engineering

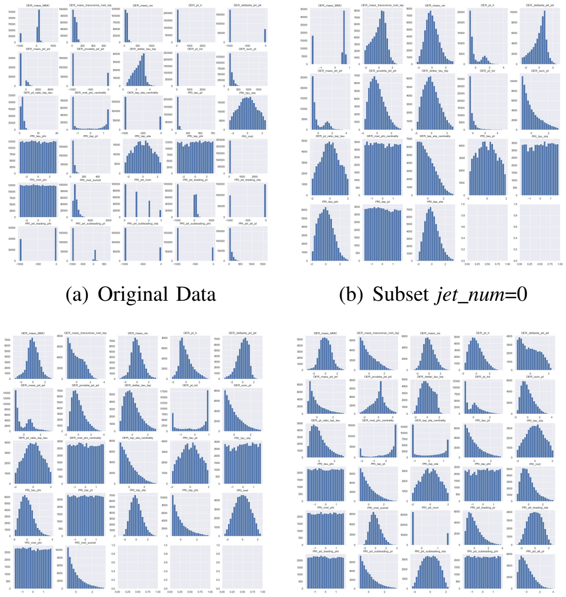
Method	degree=1	degree=3
No Regroup + One-hot	74.44%	78.37%
Regroup	75.22%	78.83%
Regroup + Norm	70.85%	79.24%
Regroup + Expansion	67.32%	67.01%
Regroup + Norm + Expansion	80.94%	79.31%

Table I
Data processing methods comparison

In this section, we elaborate our data preprocessing methods, and provide detailed comparison of cross-validation(fold = 5) accuracy of different methods as shown in table I. Here we use ridge regression as baseline model.

A. Data Regrouping

By taking a glance into the data values, we found there are many missing values as ”-999.0” or ”0” existing in the dataset. Initially we use mean values for filling them. There is a column *PRI_jet_num* with only discrete int values from {0,1,2,3}, which we initially use one-hot [2] encoding to represent it as categorical values. Reasonable at first, we soon notice that the missing values are simply not applicable with the certain *PRI_jet_num*. Based on this, we divide the original dataset into three subgroups according to *PRI_jet_num* values as $num = 0$, $num = 1$ and $2 \leq num \leq 3$. Then, as is consistent with our observation, each subgroup contains certain columns with only missing values or consistent values, and we delete these columns in each subgroup as they simply contain no useful information.



(a) Original Data

(b) Subset *jet_num*=0

(c) Subset *jet_num*=1

(d) Subset *jet_num*={2,3}

Figure 1. (a). the distribution of each feature in the original data. (b)-(d). Data distribution of features (useless features have been deleted) in each of the data subsets after data processing.

B. Normalization and Outliers

We visualize the data distribution of each features as in 1(a), which reveals that many features have values concentrated in a certain range. Besides the effect of missing values (-999,0), it is also because of outliers and categorical features. After regrouping data to eliminate missing values, we still need (1). Data normalization as $x = \frac{x-mean(x)}{std(s)}$

(2). Removing outliers, where we use IQR (Interquartile Range) to define the outliers. Specifically, any data that lies out of the range $[Q1 - 1.5 \times IQR, Q3 + 1.5 \times IQR]$ are removed from our processed dataset. The processed data are visualized as 1(b) to 1(d), where data distribution shows more reasonable patterns.

C. Feature Expansion

As our models are based on linear hypothesis, we expand the polynomial degree of every features to provide non-linear information. As polynomial degree number and regularization term weight jointly control the model's generalization capability, we treat the degree number as hyper-parameter and optimize it using grid search with range $[1, 14]$.

Beside polynomial expansion, as many of these features are derived as physical parameters, like *PRItauphi* represents angles of "hadronic tau", we also apply other operators such as trigonometric function as $sin(x)$ and $cos(x)$, absolute log as $log(|x|)$, and absolute square root as $\sqrt{|x|}$. Finally, to allow feature interactions with simple multiplications, we also explicitly add feature mutual multiplication terms $x_i x_j, i, j \in [0, N]$.

III. Experiments

A. Models and Methods

Model	γ	λ	degree	loss	test accuracy
Least Square GD	1e-6	-	2	0.499	0.726
Ridge Regression	-	1e-4	5	0.582	0.807
Logistic Regression	1e-7	1e-4	1	0.505	0.758
Least Squares	-	-	3	0.523	0.793

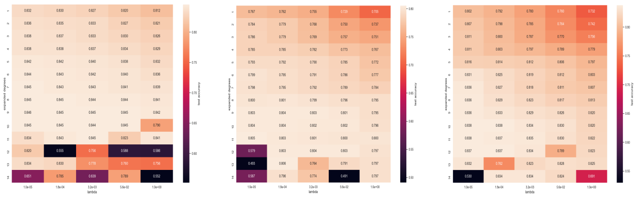
Table II
Model comparison

We implemented four basic machine learning models, and compare their performance under the best hyper-parameter settings. The results are reported in table II. We set up each model as follows:

1) Cross Validation: We used 5-fold cross validation for our model evaluation, which we first divided the whole training set into 5 subsets and used each of them for testing and the other for training. Finally, we computed the average accuracy and loss as our model performance index.

2) Grid Search: In order to find the best hyper-parameters for our model, we combined grid search and cross validation to obtain the best set of hyper-parameters, which includes the learning rate γ, regularization weight λ and polynomial expansion degree number. For model comparison, we set the maximum degree to be less than 5 thus avoiding additional computation cost.

It is worth noting that, during our experiment using Logistic regression model, we encountered "Nan" gradient that makes model not applicable. It could be solved by using a smaller learning rate with more iterations which mades logistic models time-consuming. Therefore, we choose to use the ridge regression model that shares similar results with logistic model but requires less time.



(a) *jet_num*=0 (b) *jet_num*=1 (c) *jet_num*={2,3}

Figure 2. Grid search results of test accuracy on each of the regrouped data subset according to the jet_num feature. We treat polynomial degree number and regularization weight γ as trade-off hyper-parameters

B. Hyper-parameter Settings

To obtain the best test performance and avoid model overfitting, we need to find the best degree and λ trade-off. Here we apply grid search over the two hyper-parameters from the pre-defined range degree $\in [1, 14]$ and $lambda \in$ np.logspace(-5, 0, 5). We use heatmap to efficiently visualize the relation between test accuracy and the two hyper-parameters, as shown in Figure 2. The results meet our observation that either large degree number with low γ or small degree number with high γ lead to worse performance, and the best tradeoff lies in the middle of the heatmap. We independently train ridge regression models for each of the data subset and choose the best hyper-parameters separately from the grid-search results. Accordingly, test set is also divided into three groups following the same scheme as training, and we conduct prediction using the corresponding model.

C. Results

By utilizing the grid-searched best hyper-parameters, we train three ridge-regression models for each of the divided data subset, and separately predicting the results for the divided testing sets. As a result, our model achieves **83.1%** accuracy on the public test set of HiggsML.

IV. Conclusion

Through this dataset challenge we apply ML models to the binary classification task. Our method iterated many generations and gradually leads to the final performance. We compared several ML models at first and choose the ridge-regression for efficient training. We initially processed the dataset following the standard procedure, but after taking a deeper look at the missing value and dataset background, we divided the dataset and applied feature expansion techniques, which greatly boosted the model performance. To deal with the model accuracy and generalization ability trade-off, we applied grid-search which helped our model reach the best result.

References

[1] C. Adam-Bourdarios, G. Cowan, C. Germain, I. Guyon, B. Kegl, and D. Rousseau, "Learning to discover: the higgs boson machine learning challenge," 05 2014.

[2] K. Potdar, T. Pardawala, and C. Pai, "A comparative study of categorical variable encoding techniques for neural network classifiers," *International Journal of Computer Applications*, vol. 175, pp. 7–9, 10 2017.